CLAIMS

1. A computer processor that executes instructions and that supports virtual machine monitor operation and implementation, the computer processor comprising:

a virtualization-mode processor state for execution of non-virtual-machine-monitor instructions;

a non-virtualization-mode processor state for execution of virtual machine monitor instructions; and

a virtualization-mode-switch instruction that switches the state the processor between virtualization-mode and non-virtualization-mode without incurring an interruption.

2. The computer processor of claim 1 wherein, when executed by a process running in virtualization mode, the virtualization-mode-switch instruction checks to ensure that the page attributes of the virtual memory page containing the virtualization-mode-switch instruction are compatible with the virtualization-mode-switch instruction and that the current priority level is a most privileged virtualized priority level, before switching to non-virtualization mode.

3. The computer processor of claim 1 wherein, when executed by a process running in non-virtualization mode, the virtualization-mode-switch instruction checks to ensure that the page attributes of the virtual memory page containing the virtualization-mode-switch instruction are compatible with the virtualization-mode-switch instruction and that the current priority level is a highest priority level, before switching to virtualization-mode.

4. A computer processor that executes instructions and that supports virtual machine monitor operation and implementation, the computer processor including:

a virtualization-mode processor state for execution of non-virtual-machine-monitor instructions;

a non-virtualization-mode processor state for execution of virtual machine monitor instructions; and a virtualization fault invoked by the computer processor when a routine executing in virtualization mode at a highest privilege level attempts to execute an instruction needing virtualization.

5. The computer processor of claim 4 wherein the virtualization fault has a lower priority than improper-instruction-related faults, so that an associated virtualization-fault handler in a computer system including the computer processor can avoid emulating improper-instruction-related faults.

6. The computer processor of claim 4 wherein the processor is switched to non-virtualization mode when the virtualization fault is triggered by the enhanced computer processor.

7. A computer processor that executes instructions and that supports virtual machine monitor operation and implementation, the computer processor including

a virtualization-mode processor state for execution of non-virtual-machine-monitor instructions;

a non-virtualization-mode processor state for execution of virtual machine monitor instructions; and

a flexible highest-implemented-virtual-address bit that, in virtualization mode, is checked by the processor and reported by the virtual machine monitor to be less than the highest-implemented-virtual-address bit in non-virtualization mode, so that a high-order portion of virtual-address space is accessible only to a virtual machine monitor executing in non-virtualization mode.

8. A method for enhancing a computer processor to support virtual-machine-monitor operation and implementation, the method comprising:

to a computer processor that executes instructions and that provides registers, adding

a virtualization-mode processor state for execution of non-virtual-machine-monitor instructions;

a non-virtualization-mode processor state for execution of virtual machine monitor instructions; and

a virtualization-mode-switch instruction that switches the state the processor between virtualization-mode and non-virtualization-mode without incurring an interruption.

9. The method of claim 8 further comprising, when the virtualization-mode-switch instruction is executed by a process running in virtualization mode:

checking to ensure that the page attributes of the virtual memory page containing the virtualization-mode-switch instruction are compatible with the virtualization-mode-switch instruction; and

checking that the current priority level is a most privileged virtualized priority level before switching to non-virtualization mode.

10. The method of claim 9 further comprising, when the virtualization-mode-switch instruction is executed by a process running in non-virtualization mode:

checking to ensure that the page attributes of the virtual memory page containing the virtualization-mode-switch instruction are compatible with the virtualization-mode-switch instruction; and

checking that the current priority level is a highest priority level, before switching to virtualization mode.

11. A method for enhancing a computer processor to support virtual-machine-monitor operation and implementation, the method comprising:

to a computer processor that executes instructions and that provides registers, adding

a virtualization-mode processor state for execution of non-virtual-machine-monitor instructions;

a non-virtualization-mode processor state for execution of virtual machine monitor instructions;

a virtualization fault; and

a corresponding virtualization-fault handler that is invoked by the computer processor when a routine executing in virtualization mode at the highest privilege level attempts to execute an instruction that needs virtualization

12. The method of claim 11 wherein the virtualization fault has a lower priority than improper-instruction-related faults, so that the virtualization-fault handler can avoid emulating improper-instruction-related faults.

13. The method of claim 12 further comprising switching the processor to non-virtualization mode when the virtualization fault is triggered by the computer processor.

14. A method for enhancing a computer processor to support virtual-machine-monitor operation and implementation, the method comprising:

to a computer processor that executes instructions and that provides registers, adding

a virtualization-mode processor state for execution of non-virtual-machine-monitor instructions;

a non-virtualization-mode processor state for execution of virtual machine monitor instructions; and

a flexible highest-implemented-virtual-address bit that, in virtualization mode, is checked by the processor and reported by the virtual machine monitor to be less than the highest-implemented-virtual-address bit in non-virtualization mode, so that a high-order portion of virtual-address space is accessible only to a virtual machine monitor executing in non-virtualization mode.

15. A method for supporting multiple, concurrent guest operating systems in a computer system, the method comprising:

providing a virtual-mode bit flag in a processor;

providing a *vmsw* instruction in the processor that changes the state of the virtualization-mode bit flag to enable a guest operating system to directly enter virtual-machine-monitor mode without incurring an interruption;

providing a virtualization fault with an interruption vector, assigned to have a relatively low interruption priority, that is generated when a routine, executing at high priority in virtualization mode, attempts to execute a privileged instruction or an instruction that needs software virtualization assistance and that is associated with a virtualization fault handler; and

providing a virtual-machine monitor that executes privileged instructions on behalf of guest operating systems and provides to each guest operating system a virtual machine interface, the virtual-machine monitor invoked by *vmsw* instructions or a virtualization fault.

16. The method of claim 15 wherein address space is reserved for exclusive use by the virtual-machine monitor.

17. The method of claim 16 wherein the address space reserved for exclusive use by the virtual-machine monitor is addressed by addresses including a highest bit of implemented virtual address space, the virtual-machine monitor reporting a smaller implemented address space to guest operating systems addressed by addresses that do not include the highest bit of implemented virtual address space.

18. Computer instructions stored in a computer-readable medium that implement the virtual-machine monitor of claim 15.

19. Computer instructions that implement the virtual-machine monitor of claim 15 encoded in a digital data-transfer medium including one of:

an electronic data transfer medium; and

an optical data transfer medium.